IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re the Application of:	
Daniel M. WONG et al.	Group Art Unit: 2131
Serial No.: 10/600,388	Examiner: Jackson, Jenise E.
Filed: June 20, 2003	Confirmation No. 8538
For: METHOD AND APPARATUS FOR ENABLING DATABASE PRIVILEGES	

AMENDMENT AND RESPONSE TO OFFICE ACTION

Mail Stop Amendment
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir:

In response to the Office Action mailed July 2, 2007 , please amend this application in accordance with the following amendment sheet(s).

Amendments to the Claims begin on page 2.

Remarks begin on page 7.

AMENDMENTS TO THE CLAIMS

Please amend claims 1-2, 14-15, and 26-27 as set forth below, without acquiescence in the Office Action's reasons for rejection or prejudice to pursue in a related application. A complete listing of the pending claims is provided below.

1. (Currently Amended) A method for enabling privileges comprising:

establishing a session on behalf of a user;

receiving a request to enable database privileges for the user;

upon receipt of the request to enable database privileges, verifying trusted security logic has previously been executed ~~prior to receiving the request to enable database privileges~~, wherein the act of verifying the trusted security logic comprises verifying a proxy user and checking a call stack; and

enabling database privileges for the user if the trusted security logic has previously been executed ~~prior to receiving the request to enable the database privileges~~.

2. (Currently Amended) The method of claim 1, further comprising:

storing call information in one or more frames of ~~a~~ the call stack; and wherein

the act of verifying further comprises determining whether the one or more frames of the call stack corresponds to the trusted security logic.

3. (Previously presented) The method of claim 1, wherein the act of verifying the trusted security logic further comprises verifying an application name.

4. (Original) The method of claim 3, wherein the act of verifying the trusted security logic further includes verifying a security function name.

5. (Previously presented) The method of claim 1, wherein the act of verifying trusted security logic further comprises verifying a module name.

6. (Original) The method of claim 1, further comprising:

collecting one or more session parameters;

comparing the one or more session parameters against a set of trusted security parameters defined in a security function; and

returning a result indicating whether the one or more session parameters matches the set of trusted security parameters.

7. (Cancelled)

8. (Original) The method of claim 1, further comprising:

receiving information identifying the user;

prompting the user for a password;

authenticating the user based on information stored in an application program; and

associating the user with a role.

9. (Previously presented) A client-server computer system comprising:

a computer including:

a processor,

a main memory communicatively coupled to the processor; and

a disk storage communicatively coupled to the processor;

a database running on the computer from the main memory, the database further comprising:

one or more data structures stored in the disk storage, and

a call stack stored in the main memory;

an application program coupled to the database and configured to support a user; and

a metadata repository embodied in the one or more data structures stored in the disk storage, the metadata repository comprising trusted security logic; wherein

the application program is configured to initiate a call to enable database privileges, the call causing information to be stored in one or more frames of the call stack and one or more security functions to be executed; and wherein

the database is configured to:

verify the call stack comprises one or more frames corresponding to the trusted security logic;

test a proxy user; and

enable database privileges for the user if the trusted security logic is contained in the one or more frames of the call stack.

10. (Original) The client-server computer system of claim 9, wherein the application program resides with the database in the computer.

11. (Original) The client-server computer system of claim 9, wherein the application program resides on a separate computer communicatively coupled to the database.

12. (Original) The client-server computer system of claim 9, wherein the trusted security logic includes a schema name and a security package name.

13. (Cancelled)

14. (Currently Amended) A computer-readable medium having stored therein one or more sequences of instruction for enabling privileges, the one or more sequences of instructions causing one or more processors to perform a number of acts, said acts comprising:

establishing a session on behalf of a user;

receiving a request to enable database privileges for the user;

upon receipt of the request to enable database privileges, verifying trusted security logic has previously been executed ~~prior to receiving the request to enable database privileges~~, wherein the act of verifying the trusted security logic comprises verifying a proxy user and checking a call stack; and

enabling database privileges for the user if the trusted security logic has previously been executed ~~prior to receiving the request to enable the database privileges.~~

15. (Currently Amended) The computer-readable medium of claim 14, further comprising:

storing call information in one or more frames of the ~~a~~ call stack; and wherein

the act of verifying further comprises determining whether the one or more frames of the call stack corresponds to the trusted security logic.

16. (Previously presented) The computer-readable medium of claim 14, wherein the act of verifying the trusted security logic further comprises verifying an application name.

17. (Original) The computer-readable medium of claim 16, wherein the act of verifying the trusted security logic further includes verifying a security function name.

18. (Previously presented) The computer-readable medium of claim 14, wherein the act of verifying trusted security logic further comprises verifying a module name.

19. (Original) The computer-readable medium of claim 14, further comprising:

collecting one or more session parameters;

comparing the one or more session parameters against a set of trusted security parameters defined in a security function; and

returning a result indicating whether the one or more session parameters matches the set of trusted security parameters.

20. (Cancelled)

21. (Original) The computer-readable medium of claim 14, further comprising:

receiving information identifying the user;

prompting the user for a password;

authenticating the user based on information stored in an application program; and

associating the user with a role.

22-25. (Cancelled)

26. (Currently Amended) A system for enabling privileges comprising:

means for establishing a session on behalf of a user;

means for receiving a request to enable database privileges for the user;

means for upon receipt of the request to enable database privileges, verifying trusted security logic has previously been executed ~~prior to receiving the request to enable database privileges~~, wherein means for verifying the trusted security logic comprises means for verifying a proxy user and checking a call stack; and

means for enabling database privileges for the user if the trusted security logic has previously been executed ~~prior to receiving the request to enable the database privileges~~.

27. (Currently Amended) The system of claim 26, further comprising:

means for storing call information in one or more frames of ~~a~~ the call stack; and wherein

means for verifying further comprises means for determining whether the one or more frames of the call stack corresponds to the trusted security logic.

28. (Previously presented) The system of claim 26, wherein means for verifying the trusted security logic further comprises means for verifying an application name.

29. (Previously presented) The system of claim 28, wherein means for verifying the trusted security logic further comprises means for verifying a security function name.

30. (Previously presented) The system of claim 22, wherein means for verifying trusted security logic further comprises means for verifying a module name.

31. (Previously presented) The system of claim 22, further comprising:

means for collecting one or more session parameters;

means for comparing the one or more session parameters against a set of trusted security parameters defined in a security function; and

means for returning a result indicating whether the one or more session parameters matches the set of trusted security parameters.

32. (Previously presented) The system of claim 22, further comprising:

means for receiving information identifying the user;

means for prompting the user for a password;

means for authenticating the user based on information stored in an application program; and

means for associating the user with a role.

REMARKS

Claims 1-6, 8-12, 14-19, 21 and 26-32 are currently pending in the application. In the Office Action dated July 2, 2007, claims 1-6, 8-12, 14-19, 21 and 26-32 were rejected. By this Amendment, claims 1-2, 14-15 and 26-27 have been amended, without acquiescence or prejudice to pursue the original claims in a related application. No new matter has been added.

Claim Rejections - 35 USC § 102

A. Claims 1-6, 9-12, 14-19, and 26-31, are rejected under 35 U.S.C. 102(e) as being anticipated by Bernstein et al. (5,884,316).

1. Claim 1 recites the limitation "upon receipt of the request to enable database privileges, verifying trusted security logic has previously been executed, wherein the act of verifying the trusted security logic comprises verifying a proxy user and checking a call stack." Claims 9, 14, and 26 have similar limitations. Applicants respectfully submit that Bernstein does not disclose verifying trusted logic has previously been executed by verifying a proxy user and checking a call stack **as claimed**.

Bernstein is directed toward eliminating the need to pass session information explicitly in each server call by implicitly associating each object in the server with a session (Bernstein, Column 2, lines 30-65). Bernstein associates each object in the server with a session such that every server call automatically runs with the session identifier of the object called (Bernstein, Column 2, line 30).

However, Bernstein is completely silent with respect to verifying trusted logic by checking a call stack. Bernstein associates all objects at the server with a session identifier to ensure there is no need to explicitly pass session information. Therefore, verification of session information is unnecessary because all objects in Bernstein have session information and a check of the call stack would not allow for verification of session information. Thus, Bernstein does not disclose verifying trusted logic has previously been executed by checking a call stack.

Moreover, Bernstein does not disclose verifying trusted logic has previously been executed by verifying a proxy user. According to the Office Action, Column 2, lines 50-67,

Column 5, lines 50-55, and Column 6, lines 24-36 of Bernstein discloses "verifying trusted security logic has been executed prior to receiving the request to enable database privileges, wherein the act of verifying the trusted logic includes verifying a proxy user." Applicants note that the cited section of Bernstein merely disclose the general concept of objects in conjunction with database structures, and fails to disclose, suggest, or describe any information about proxy users, much less verifying that trusted logic has previously been executed by verifying a proxy user as claimed.

For at least these reasons, Applicants submit that Bernstein fails to disclose, teach or suggest every limitation of claim 1. Because claims 9, 14, and 26 have similar limitations to claim 1 discussed above, they are not anticipated by Bernstein. Furthermore, because claims 2-6, 10-12, 15-19, and 27-31 depend from claims 1, 9, 14, and 26, they also are not anticipated by Bernstein.

Claim Rejections - 35 USC § 103

A. Claims 8, 21, and 32, are rejected under 35 U.S.C. 103(a) as being unpatentable over Bernstein (5, 884,316) in view of Fisher et al. (6,092,189).

1. As discussed above, Bernstein fails to disclose the limitations in independent claims 1, 14, and 26 from which claims 8, 21, and 32 depend. Fisher does not remedy the deficiencies present in Bernstein. Fisher discloses a process for mass production of computers with automatic installation of software and does not disclose verifying trusted logic has previously been executed by verifying a proxy user and checking a **call stack**.

For at least these reasons, Applicants submit that Bernstein in view of Fisher fails to disclose, teach or suggest every limitation of claim 1. Because claims 9, 14, and 26 have similar limitations to claim 1 discussed above, they are not anticipated by Bernstein in view of Fisher. Furthermore, because claims 2-6, 8, 10-12, 15-19, 21, and 27-32 depend from claims 1, 9, 14, and 26, they also are not anticipated by Bernstein in view of Fisher.

CONCLUSION

Based on the foregoing, all claims are believed allowable, and an allowance of the claims is respectfully requested. If the Examiner has any questions or comments, the Examiner is respectfully requested to contact the undersigned at the number listed below.

If the Commissioner determines that additional fees are due or that an excess fee has been paid, the Patent Office is authorized to debit or credit (respectively) Deposit Account No. **50-4047**, billing reference no. **7011293001**.

Respectfully submitted,

Dated: December 3, 2007

By: _____

Erich C. Tzou
Registration No.: 56,927
for
Peter C. Mei
Registration No.: 39,768

Bingham McCutchen LLP
Three Embarcadero Center
San Francisco, CA 94111-4067
Telephone: (650) 849-4400
Telefax: (650) 849-4800